
Operator Native Thinking

# ONT Platform

## Infrastructure has a Memory

*"Operations has a kind of pain that rarely gets named. It lives in fragmented logs, lost context, midnight debugging, and decisions that made sense in the moment but cannot be explained later. It is not just technical debt. It is memory loss at scale."*

*"The cluster is the documentation. Not a representation of it. Not a mirror. The thing itself. With an API that makes the organization queryable for the first time."*

A living documentation system and infrastructure governance framework for on-premises Kubernetes. Declarative. Versioned. Auditable. Human-at-boundary.

Krishna | SRE | Platform Engineer | Infrastructure Governance Engineer

ontai.dev | April 2026

# 0. The Cluster Is the Documentation

Every engineering organization maintains two systems simultaneously. The running infrastructure: precise, machine-executable, continuously changing. The documentation of that infrastructure: human-readable, manually maintained, and drifting from reality the moment it is published. These two systems have never been the same thing.

***ONT collapses this gap entirely. Not by improving the representation. By eliminating the need for one.***

When every organizational decision is a CRD, every policy a versioned resource, every contract a reconciled object, the cluster is the documentation. The CRD is not a document about the payment rail authorization. It is the authorization. The operator is not a system that implements the runbook. It is the runbook, running.

## Five Things That Become Cluster Properties

| | |
|---|---|
| **System topology** | Query the dependency graph. Receive the current, live, reconciled topology as structured data. No architecture diagram required. |
| **Compliance posture** | Query the governance layer of relevant CRDs. Receive declared regulatory constraints as typed fields. No compliance register spreadsheet required. |
| **Change history** | Query the governance event chain. Receive the sequence of human decisions that produced the current state, with attribution and rationale. |
| **Dependency graph** | Traverse declared typed references between CRDs. Receive the complete dependency topology as a live graph. No stale diagram required. |
| **Audit trail** | Query the structured CNPG audit sink. Receive attributed, timestamped, lineage-traced records of every governance event. No log aggregation required. |

## The Operator at 3am

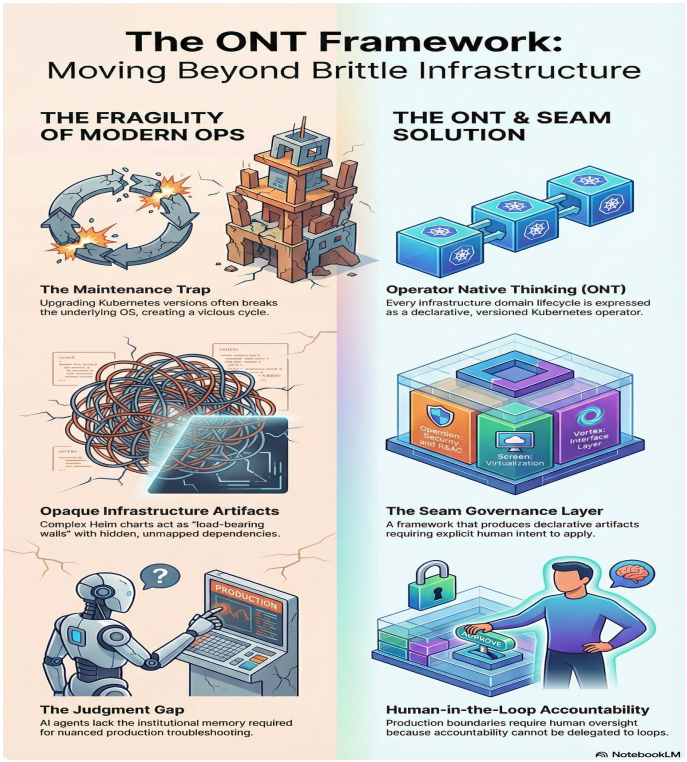
With traditional documentation, the operator at 3am begins archaeology. They open the wiki. They find a page last updated eight months ago. The topology does not match. They search Slack. They piece together context from threads and git blame.

With ONT, the operator queries the cluster. The answers are not representations. They are the truth. The cluster holds the truth because the operator model enforces it, reconciles toward it, and records every deviation from it.

***Infrastructure governance is the consequence of living documentation, not the purpose.***

# 1. The Problem: Infrastructure Without Memory

Modern infrastructure operations are held together by accumulated tribal knowledge, manual interventions, and undocumented decisions. When the engineer who made a critical choice leaves, the reasoning leaves with them. When a production incident strikes at midnight, operators piece together context from logs that were never designed to be read together.




*The fragility of modern ops and the ONT solution*

## Three Root Causes

**The Maintenance Trap.** Upgrading Kubernetes breaks the underlying OS. Upgrading the OS breaks the workloads. Organizations get locked into a perpetual cycle of deferred upgrades, each one more dangerous than the last.

**Opaque Infrastructure Artifacts.** Complex Helm charts accumulate over years into load-bearing walls with hidden dependencies. Nobody fully understands them. Removing a single value can bring down production. The knowledge of why those values exist is long gone.

**The Judgment Gap.** AI agents today lack the institutional memory required for nuanced production troubleshooting. They can execute. They cannot remember why the last execution failed, what the human decided as a result, and how that decision reshaped the system.

# 2. Why AI is Not Yet Ready for Production Operations

The infrastructure industry is racing to inject AI into production operations. The promise is compelling: autonomous remediation, intelligent scaling, self-healing systems. The reality is more sobering. Production operations require something that current AI systems fundamentally lack: durable, structured, auditable memory.

> ***"AI does not fail because it lacks intelligence. It fails because it lacks memory. And in operations, memory is accountability."***

## The Five Gaps

| | |
|---|---|
| 1 | **No Causal Memory**<br>Current LLMs have no persistent memory of past decisions. Each invocation starts fresh. In production, every action has a history and every decision has a context. An AI that cannot recall why a circuit breaker was tripped last Tuesday cannot safely touch it today. |
| 2 | **No Accountability Chain**<br>Production changes require an audit trail: who decided what, why, and what the outcome was. Current AI systems cannot produce a verifiable, tamper-evident record of their own decision path. Regulators, auditors, and post-mortems require this. |
| 3 | **No Domain Context**<br>Infrastructure decisions are embedded in organizational context: business priorities, SLAs, past incidents, team conventions. An AI trained on public data carries none of the institutional knowledge of the specific organization it is operating in. |
| 4 | **Hallucination at the Worst Moment**<br>Production incidents are exactly the high-pressure, time-critical, ambiguous situations where LLMs are most likely to hallucinate plausible-sounding but incorrect remediation steps. The cost of a confident wrong answer in production is catastrophic. |
| 5 | **No Human Approval Gate**<br>The most dangerous assumption in autonomous AI operations is that the AI knows when to stop and ask. Current systems have no principled, enforced boundary between safe autonomous action and decisions that require human judgment. Accountability cannot be delegated to a reconciliation loop. |

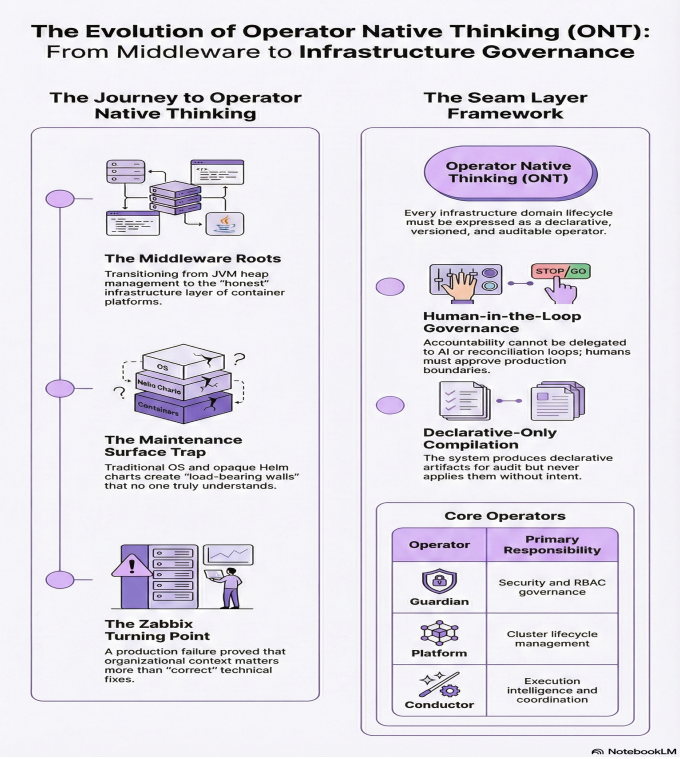
## The ONT Position on AI

ONT does not reject AI. It defines the correct role for AI in infrastructure operations. AI belongs in the R and D acceleration layer: generating operator session prompts, suggesting architectural improvements,

drafting schema declarations for human review. AI never touches production directly. The human reviews the manifest. The human applies it. The platform records that the human applied it. This is not a limitation. This is the architecture of trust.

| AI Role | Safe | Rationale |
|---|---|---|
| Generate Claude Code session prompts | **Yes** | Human reviews and applies output |
| Draft schema declarations | **Yes** | Compiler validates, human commits to git |
| Suggest architectural improvements | **Yes** | Governor session, human locks decision |
| Autonomous production remediation | **No** | No audit trail, no causal memory |
| Apply infrastructure changes without review | **No** | Accountability cannot be delegated to loops |
| Make upgrade decisions autonomously | **No** | Requires organizational context AI cannot hold |

# 3. The ONT and Seam Solution




*The evolution from middleware roots to infrastructure governance*

ONT (Operator Native Thinking) is the methodology that restructures infrastructure operations from a collection of manual procedures and tribal knowledge into a versioned, declarative, auditable governance system. Every infrastructure domain lifecycle is expressed as a Kubernetes operator. Every decision is recorded. Every change is traced.

## Three Governing Principles

| | |
|---|---|
| **Declarative** | Desired state lives in etcd as a structured record, not as a runtime instruction. Operators reconcile against declared intent. Nothing happens by running a script. Everything happens by applying a manifest. |
| **Traced** | Every CRD carries a mandatory lineage reference to its domain authority. The InfrastructureLineageIndex tracks every infrastructure object. The SealedCausalChain makes lineage immutable once set. |
| **Auditable** | Every state transition is timestamped, attributed to an actor, and written to the Guardian CNPG audit sink. The audit trail is the authoritative record of what happened, when, and why. |

# 4. The Seam Operator Family

Six operators form the Seam infrastructure domain. Each owns a distinct governance surface. No operator reimplements another. All are governed by Guardian.

### Guardian

security.ontai.dev

Trust root and security substrate. Owns RBAC governance, PermissionSnapshot freshness, identity resolution, and the CNPG audit sink. Every operator is admitted to the Seam family through Guardian validation. Guardian runs in management and tenant roles without code changes.

**CRDs:** RBACPolicy, RBACProfile, PermissionSnapshot, IdentityBinding, IdentityProvider, SeamMembership (via seam-core)

### Platform

platform.ontai.dev

Cluster lifecycle authority. Imports and bootstraps Talos Linux clusters. Manages CAPI-driven cluster provisioning via SeamInfrastructureCluster and SeamInfrastructureMachine. Auto-onboards tenant clusters: RBACPolicy, RBACProfiles, LocalQueue, kubeconfig.

**CRDs:** TalosCluster, RunnerConfig, SeamInfrastructureCluster, SeamInfrastructureMachine, TalosControlPlane, TalosWorkerConfig

### Wrapper

infra.ontai.dev

Pack delivery engine. Manages the full lifecycle of OCI artifact packs from ClusterPack signing through PackExecution gate clearing to PackInstance creation and DNS registration. Creates PackExecution directly from ClusterPack without ephemeral intermediates.

**CRDs:** ClusterPack, PackExecution, PackInstance

### Conductor

runner.ontai.dev

Execution intelligence. Two binaries: Compiler (never deployed, produces manifests for human review) and Conductor (distroless, deployed everywhere). Owns capability publishing, ClusterPack signing loop, and pack-deploy Job execution with staged apply.

**CRDs:** RunnerConfig status, OperationResult ConfigMap, 17 capability handlers

### seam-core

infrastructure.ontai.dev

| | |
|---|---|
| Lineage and semantic DNS controller. Maintains InfrastructureLineageIndex across all nine operator GVKs. Owns SeamMembership CRD. Operates the DSNSReconciler that writes the authoritative seam.ontave.dev zone. | **CRDs:** InfrastructureLineageIndex, SeamMembership, DSNS zone (SOA, A, TXT records) |

## Screen [ROADMAP]

screen.ontai.dev

| | |
|---|---|
| Virtualization infrastructure provider. Extends the TalosCluster CAPI path to provision QEMU/KVM virtual clusters via libvirt and KubeVirt. Enables Seam governance over virtual infrastructure in addition to bare-metal. Currently in roadmap. | **CRDs:** VirtCluster, VirtMachine (roadmap) |

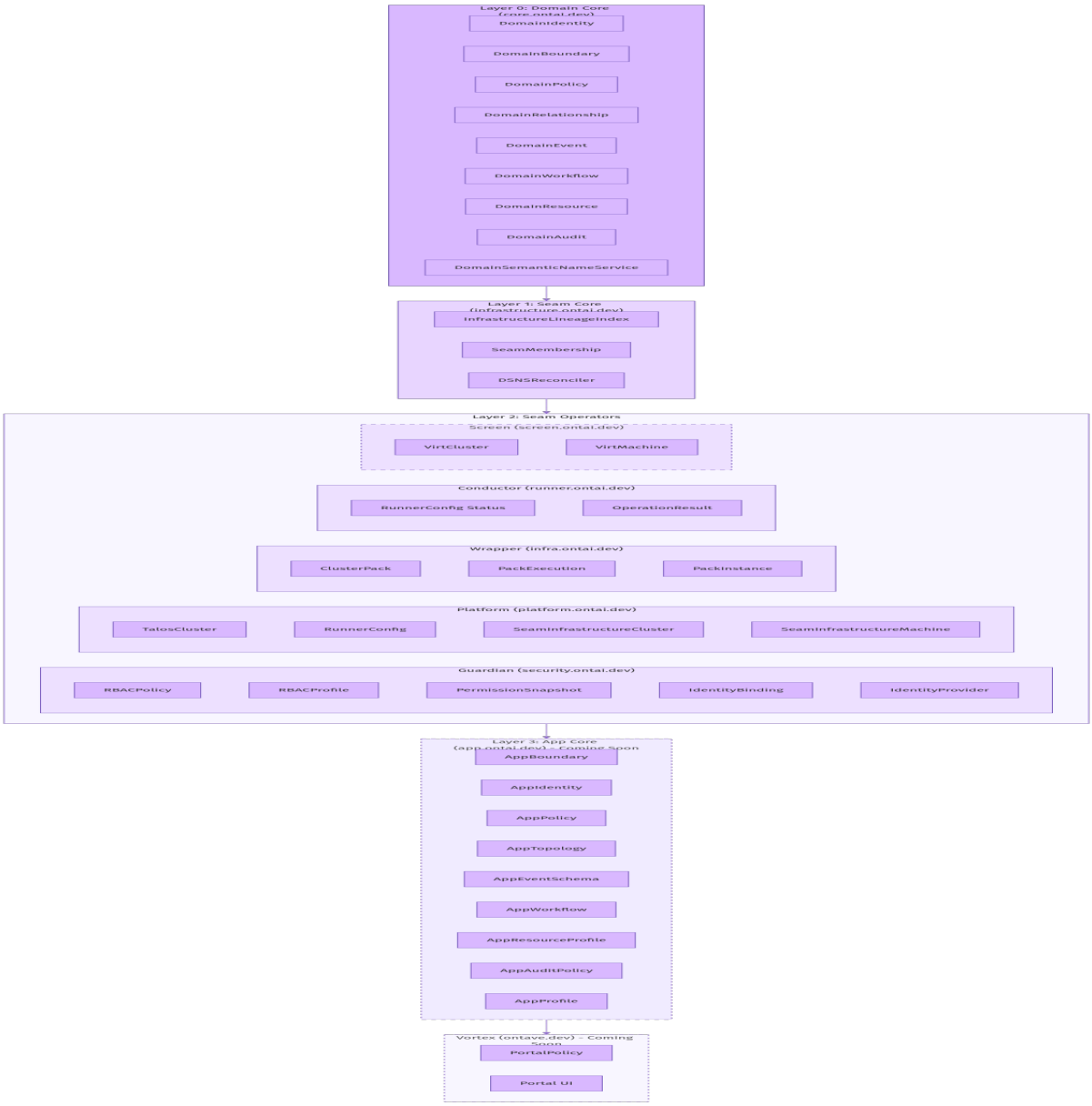
## ONTAR [ROADMAP]

runtime.ontai.dev

| | |
|---|---|
| NOT IMPLEMENTED. Future specification only. Operator Native Task Application Runtime. Extends the governance chain from cluster-level policy to pod-level execution contract. The Go binary as PID 1, no shell, ephemeral runtime CA per pod, child runtime declares intent rather than executing commands directly. | **CRDs:** PermissionSnapshot pod contract (future spec) |

# 5. CRD Architecture and Layer Model

The platform is organized into four strict layers. Each layer depends on the layer above it. No layer reimplements the layer above it. The dependency flows strictly downward.




*ONT four-layer CRD architecture: domain-core, seam-core, Seam operators, and application layer*

## Schema Integrity Chain

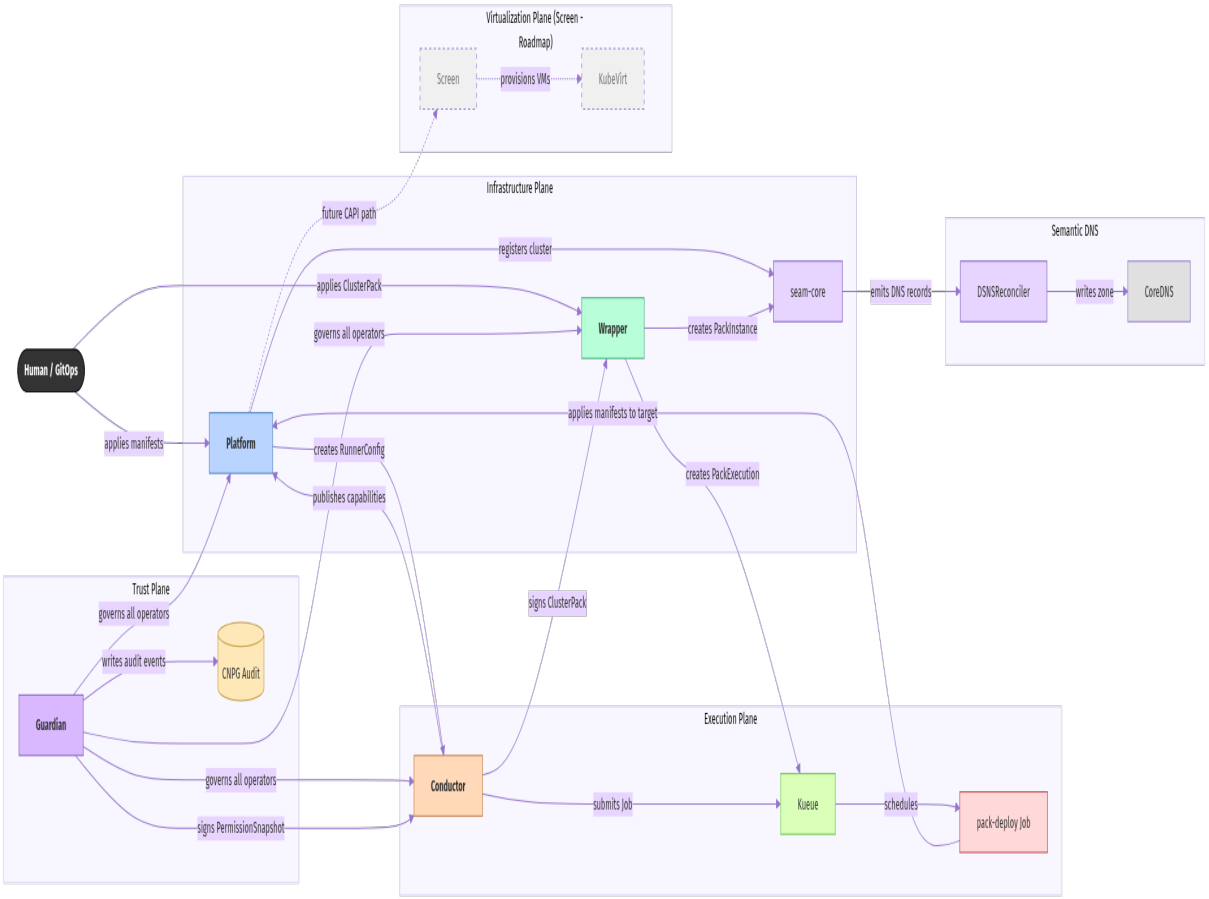
Every operator CRD traces to a domain authority through a formal lineage chain. This chain is locked in the current codebase.

| | | |
|---|---|---|
| 1 | **DomainRelationship** domain-core | 6 cross-operator relationships declared in YAML. |

| | | |
|---|---|---|
| 2 | **RBACProfile.domainIdentityRef** guardian | Traces each operator SA to its DomainIdentity at core.ontai.dev. |
| 3 | **InfrastructureLineageIndex.domainRef** seam-core | Set to "infrastructure.core.ontai.dev" by LineageController. Validated at CREATE. |
| 4 | **SeamMembership** seam-core + guardian | Guardian validates domainIdentityRef match and RBACProfile provisioned gate. |
| 5 | **PermissionSnapshot** guardian | Resolved after membership admitted. Operator formally enters Seam family. |

# 6. Operator Relationship Map

The six operators interact across three planes: trust, infrastructure, and execution. Every interaction is governed by a declared DomainRelationship in domain-core.




*Seam operator interaction planes: trust, infrastructure, execution, and semantic DNS*

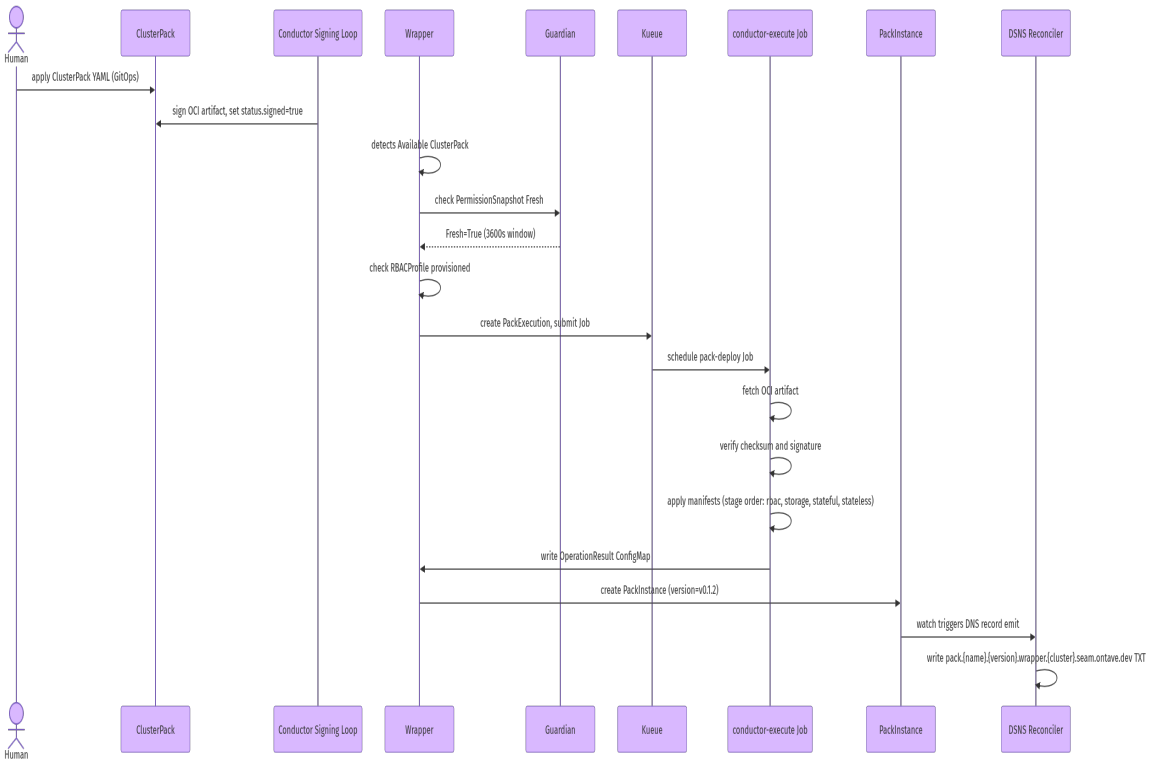
## Six Declared DomainRelationships

| Relationship | Source | Target | Type | Protocol |
|---|---|---|---|---|
| guardian-signs-conductor | guardian | conductor | signs | Ed25519 |
| conductor-signs-clusterpack | conductor | wrapper | signs | Ed25519 |
| platform-creates-runnerconfig | platform | conductor | provisions | Kubernetes CRD |

| Relationship | Source | Target | Type | Protocol |
|---|---|---|---|---|
| wrapper-creates-packexecution | wrapper | conductor | delegates | Kubernetes CRD |
| guardian-provisions-rbacprofile | guardian | all operators | governs | Kubernetes CRD |
| seam-core-tracks-lineage | seam-core | all | observes | Kubernetes watch |

# 7. Pack Delivery Flow

The pack delivery flow is the primary mechanism for delivering workloads to any cluster in the Seam domain. The human authors only the ClusterPack CR and applies it via GitOps. Everything downstream is operator-generated, gate-checked, and audit-recorded.




*Full pack delivery sequence: from GitOps apply through DNS record emission*

## Four Execution Gates

| | |
|---|---|
| **Gate 0** | **ConductorReady**<br>Conductor has published all 17 capabilities to the cluster RunnerConfig in ont-system. Verified by checking RunnerConfig status.capabilities. |
| **Gate 1** | **PackSignaturePending**<br>ClusterPack OCI artifact has been signed by the management cluster Conductor signing loop. Verified by status.signed=true. |
| **Gate 2** | **PermissionSnapshotOutOfSync**<br>PermissionSnapshot for the target cluster is Fresh (within the 3600s window). Auto-refreshes via EPG watch on Fresh condition. |
| **Gate 3** | **RBACProfileNotProvisioned**<br>The admissionProfileRef RBACProfile is provisioned=true. Set by Guardian after validating the operator membership chain. |

# 7b. The ONT Schema Specification

The ONT schema specification is the community standard for expressing domain contracts in the ONT governance model. It is published as an OpenAPI JSON Schema specification at schema.ontai.dev and importable by any community operator.

| Layer | Schemas | Purpose |
|---|---|---|
| shared | 6 schemas | SealedCausalChain, BindingStability, PhaseModel, RationaleField, GovernanceEvent, KubernetesMetadata |
| domain-core | 9 schemas | DomainIdentity, DomainBoundary, DomainPolicy, DomainRelationship, DomainEvent, DomainWorkflow, DomainResource, DomainAudit, DomainSemanticNameService |
| seam-core | 3 schemas | InfrastructureLineageIndex, SeamMembership, DSNSZone |
| app-core | 9 schemas | AppBoundary, AppIdentity, AppPolicy, AppTopology, AppEventSchema, AppWorkflow, AppResourceProfile, AppAuditPolicy, AppProfile |

```
Import any schema: https://schema.ontai.dev/v1alpha1/index.json
```

All 27 schemas are published under Apache License 2.0. Community contributions follow the operator validation framework: specification before code, senior engineer sign-off required, behavior test suite covering all named behaviors.

# 8. Path to Vortex: The Human Portal

Vortex is the human-at-boundary portal. It is the first application-layer operator and the proof of ONT philosophy: AI curates, the human decides, the platform records. Vortex is ontave.dev: the enterprise product layer built on the open-source ontai.dev foundation.

***The portal that governs application configuration is itself governed by every rule it enforces. Vortex is not exempt from the framework. It is the first and most visible consumer of it.***

## Five Prerequisites in Sequential Order

| | |
|---|---|
| 1 | **Alpha release of 5 repos**<br>Clean enable-ccs-mgmt.sh CI script. Tenant cluster onboarding confirmed end to end. No manual patches anywhere in the flow. |
| 2 | **app-core repository**<br>AppBoundary, AppIdentity, AppPolicy, AppTopology, AppEventSchema, AppWorkflow, AppResourceProfile, AppAuditPolicy, AppProfile. Released open source together with domain-core and Seam operators. |
| 3 | **Vortex SeamMembership**<br>Vortex submits AppProfile. Guardian evaluates AppPolicy ceiling. SeamMembership tier=application admitted. Seam-tier PermissionSnapshot resolved. |
| 4 | **Vortex AppTopology**<br>Three structural wirings: Guardian gRPC PermissionService (SnapshotBinding), GitOps webhook (ContinuousBinding), CNPG database (SnapshotBinding). Each traces to a DomainRelationship. |
| 5 | **Vortex UI**<br>AppProfile authoring with AI-curated manifest generation, RelationshipAdmissionRequest human approval workflow, cluster topology visualization via DSNS, drift detection surfacing. |

# 9. The Business and Investment Case

ONT addresses a gap that every enterprise with on-premises infrastructure faces but few have named: the cost of infrastructure without memory. That cost is not theoretical.

| Problem | Industry Cost | ONT Contribution |
|---|---|---|
| Undocumented production changes | Average incident resolution 4-8 hours | Every change is timestamped and attributed in CNPG audit sink |
| Helm chart complexity | 40% of ops time on upgrade risk management | Declarative ClusterPacks with staged apply and drift detection |
| Engineer turnover knowledge loss | 6-12 months to replace institutional knowledge | Lineage chain preserves intent across personnel changes |
| Compliance audit preparation | Weeks of manual log collection per audit | Audit trail is always current in structured CNPG tables |
| Autonomous AI incidents | Growing regulatory risk, no accountability chain | Human-at-boundary enforced by platform design, not policy |

## Market Position

ONT occupies a position that no existing product addresses: sovereign infrastructure governance for on-premises Kubernetes with the auditability and reliability of cloud providers. CloudOps methodology applied to bare-metal.

| Dimension | Traditional Ops | Cloud Platforms | ONT Platform |
|---|---|---|---|
| Auditability | Manual log archaeology | Cloud-provider logs (opaque) | Structured, attributable, complete |
| Change accountability | None by default | Cloud IAM (shared) | Operator-native, lineage-traced |
| AI boundary | None | Provider-defined | Human-at-boundary, enforced architecturally |
| Data sovereignty | On-premises | Cloud-dependent | Fully on-premises, open source |
| Upgrade safety | Manual, high risk | Managed, provider risk | Talos Linux, declarative, operator-controlled |

# 10. The Soul of ONT


*Operations has a kind of pain that rarely gets named.*

*It lives in fragmented logs, lost context, midnight debugging, and decisions that made sense in the moment but cannot be explained later. It is not just technical debt. It is memory loss at scale.*

*OntAI starts from that pain.*

*Even its name carries it. Ont means pain in Swedish, and that is intentional. This is not another layer of automation trying to replace the operator. It is an AI built to understand what operators go through and to stand beside them.*

*Its role is simple but powerful. It becomes the living memory of the operator. Every action, every decision, every piece of intent is captured, structured, and made traceable.*

*Not to control. Not to obscure. But to empower.*

*When the system remembers, the operator is free to think clearly. Free to focus on intent instead of reconstruction. Free to move forward instead of constantly looking back.*

*OntAI does not try to take over operations. It gives operators something they have always needed but never truly had.*

***A reliable memory they can trust.***